Surface Landmark Selection and Matching in Natural Terrain

Andrew E. Johnson
Jet Propulsion Laboratory, California Institute of Technology
Mail Stop 125-209, 4800 Oak Grove Drive, Pasadena, CA 91109

Abstract

In this paper we present an algorithm for robust absolute position estimation in natural terrain based on landmarks extracted from dense 3-D surfaces. Our landmarks are constructed by concatenating pose dependent oriented surface points with pose invariant surface signatures into a single feature vector; this definition of landmarks allows a priori pose information to be used to constrain the search for landmark matches. The first step in our algorithm is to extract landmarks from stable and salient surface patches. These landmarks are then stored in a closest point search structure with which landmarks are matched efficiently using available pose constraints and invariant values. Finally, an iterative pose estimation algorithm, based on least median squares, is wrapped around landmark matching to eliminate outliers and estimate absolute position. To validate our algorithm, we show hundreds of absolute position estimation results from three different natural scenes. These results show that our algorithm can incorporate constraints on position and attitude for efficient landmark matching and match small and dense scene surface patches to large and coarse model surfaces.

1 Introduction

Absolute position estimation determines the position of a robot in a global coordinate system by comparing data collected with onboard sensors to a stored model of the world. Often absolute position estimation is obtained by matching landmarks extracted from sensed data to landmarks stored in a global database. The defining qualities of landmarks are an invariance to rigid transformation, stability in the presence of changes in viewing direction and illumination conditions, and global descriptiveness that reduces ambiguity during matching.

In real situation, there often exists external information that can be used to guide the search for landmarks. For example, other onboard sensors (e.g., gyros) may give an initial estimate on the pose of the sensor. In another case domain knowledge is helpful; a car cannot be driven upside down. If this kind of information can be incorporated systematically into landmark search, then more efficient and robust absolute position estimation algorithms will result. To this end, we have developed a new representation for surface landmarks constructed from dense 3-D data. Our representation combines the pose of an oriented point on a sensed surface with a vector of pose independent invariants into a single feature vector. In this paper we show how, using this combined feature vector, it is possible to apply domain knowledge about the pose of the sensor in a systematic way to limit landmark search.

Our application domain is autonomous navigation in natural terrain. One of the difficulties of navigating in natural terrain is developing a model for landmarks that applies to the irregular and highly variable surfaces encountered in nature. Spin-images [6][7] have been shown to be effective invariants for matching free form surfaces, and we use them as the invariant component of our landmarks. However, if spin-images are to be used in landmarks, a way to select stable and distinct spin-images from all spin-images computed on a surface must be developed. In this paper we define spin-image saliency and show how to use it to select distinct landmarks.

Matching of high-resolution local scene patches to a coarse model of the terrain is a useful capability for autonomous navigation. This capability allows a robot to determine its position in a global sense, which can ultimately improve mission planning as well as reduce dead reckoning errors. However, the large fraction of model surface not represented in the scene and the difference in resolution between the data sets make this problem especially difficult for surface matching algorithms. To demonstrate the effectiveness of our algorithm, we present more than one hundred results from three different natural scenes where this problem is solved.

Multiple researchers have shown that surface shape data can be used directly estimate the position between surfaces. Besl and McKay [1] and Zhang [14] developed iterative closest point algorithms (ICP) for aligning surfaces. These algorithms are effective, but they require an initial estimate of the alignment between surfaces. Sharp et al. [10] augmented the traditional ICP algorithm with invariants to increase the range of convergence but their algorithm is still fundamentally local. On the other end of the spectrum are absolute position estimation algorithms that require no knowledge of the pose between surfaces [6][7][5][3]. In this paper we develop the middle ground by creating an algorithm that can seamlessly incorporate as much pose information as is available to guarantee robust and efficient landmark matching.

2 Surface Landmarks

We represent surfaces with meshes, piecewise linear 3-D surfaces composed of vertices and faces. We use surface meshes because they are proven representations for surface matching and can represent the complicated and irregular surfaces expected in natural scenes. Given a surface mesh, we can generate a landmark at any vertex using the 3-D position, surface normal and surrounding surface shape of the vertex. First we create an oriented point using the vertex position and surface normal. This oriented point defines the pose dependent component of a landmark and provides a coordinate system in which to define the invariants used in matching.

The surface invariants we use are based on spin-images. Spin-images are pose independent encodings of the local surface shape around an oriented point. Spin-images were introduced in [6] where they were applied to the problem of surface matching. Briefly, a spin-image is generated as follows. With respect to an oriented point, a 3-D point has two parameters: the distance from the normal line and the signed distance from the tangent plane defined by the oriented point. By projecting every surface point in the vicinity of an oriented point into a 2-D accumulator indexed by these parameters, an image is generated. This image is pose independent and, because of its finite support, has robustness to clutter and occlusion.

When spin-images are defined at every vertex of a mesh, a high dimensional surface invariant manifold is generated. Since this set of spin-images contains a large amount of redundant information, this manifold for the most part exists in a small dimensional sub-space of the original spin-image space. Using this observation, it was shown in [7] that spin-images can be compressed using principal component analysis (PCA) and replaced by low dimensional tuples of invariants, called a spin-tuple, without significant loss in matching fidelity.

With these definitions in hand, we can now give a precise definition of a landmark. A landmark defined at an oriented point is a feature vector generated from the concatenation of the oriented point position $\boldsymbol{p} = [p_x,p_y,p_z]$, oriented point surface normal $\boldsymbol{n} = [n_x,n_y,n_z]$, and spin-tuple $\boldsymbol{i} = [i_1,\ldots,i_t]$ that results from compression of the spin-image generated at that oriented point. If the tuple of invariants has t components then a landmark is the $t+6$ dimensional feature vector

$$(1) \quad \boldsymbol{l} = [p_x, p_y, p_z, n_x, n_y, n_z, i_1, \Lambda, i_t]^T$$

By combining pose and shape invariant information into a single vector, pose information can be used to limit the search during landmark matching. Surface position is used to limit search when bounds are placed on translation between model and scene and surface normal is used to limit search when bounds are placed on rotation between model and scene. Given these bounds, the invariant of a landmark is used to find the exact match between model and scene landmarks. The use of pose estimates to limit search for landmarks will be explained in full in Section 2.3, but first we will explain some improvements to spin-image generation and selection for efficient matching.

2.1 Landmark Generation

While sticking close to the original spin-image generation and matching algorithms, we have come up with some improvements to the original techniques that significantly increase matching accuracy and speed. To systematically show the benefit of our modifications, we have analyzed each modification by matching spin-images generated from two synthetic surface meshes. These meshes describe the same scene shape but are constructed to have different surface sampling, different mesh connectivity, and different randomly generated vertex position noise. These differences ensure that no two vertices from the two meshes correspond to exactly the same position in the scene, so corresponding spin-images generated from the different meshes will be similar but never exactly the same. To test the modifications, spin-images from one mesh were compared to spin-images from the other mesh. If the vertices corresponding to the best matching spin-images are also the closest vertices in Euclidean space, then the match was correct, otherwise the match is incorrect. For each modification, the percentage of correct matches and the match time are shown in Table 1

As a starting point we matched spin-images generated using a cylindrical parameterization. To minimize the effect of mesh resolution and variability between vertices, we use a variant of the discrete version of surface interpolation proposed in [5]. In our approach, spin-images are still generated at vertices of the original mesh. However, a (different) set of points that is guaranteed to have a uniform distribution over the surface of the mesh is used to generate the spin-images. During spin-image generation, instead of incrementing the spin-image by 1 for each point that falls within the spin-image support, the spin-image is incremented by a fixed interpolation point area. This guarantees that the spin-images encode surface area and not vertex density, which enables landmark matching between meshes of different resolutions. As shown in Table 1, using a cylindrical parameterization and surface interpolation we obtained 76.3% correct matches with matching taking 356 ms/image in our two synthetic data sets.

The next modification was to change the spin-image parameterization from cylindrical to a spherical. As shown in Figure 1, the spherical parameterization projects a 3-D point into a spin-image using the radial distance ρ and the elevation angle ϕ. The motivation behind a spherical parameterization is to reduce spin-image inconsistencies due to surface normal error. In the cylindrical parameterization, the effect of the error in surface normal on the spin-image generated will increase with the distance from the oriented point origin. This can cause the outer pixels of the spin-images to become uncorrelated which will increase the variance of the pixels and decrease spin-image correlation. However, with the spherical parameterization, the effect of surface normal error will be constant across the image. If the images are compared through correlation, this means that a constant bias will be introduced between matched pixels but the pixel variance will remain the same, so the chance of correct spin-image matching improves. Using the spherical parameterization increased the correct match percent to 80.8% while the match time is roughly the same as before at 336 ms/ image.

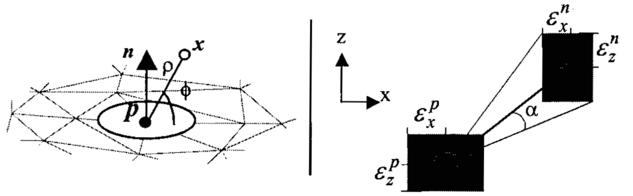


Figure 1 Spherical spin-image parameterization and hypercell bounds illustration.

2.2 Landmark Selection

The next modification was made based on the observation that many of the incorrect matches are occurring in areas of the mesh with high curvature. This seems reasonable; in areas of high curvature, surface normal cannot be computed robustly. If vertices with high curvature are eliminated from matching then the likelihood of obtaining correct matches goes up at the expense of eliminating some vertices that may have been matched correctly.

To eliminate surfaces based on curvature we first smooth the surface mesh [11] and then compute curvature at each vertex [12]. Then, for both meshes, we only generate and match spin-images at vertices that have a curvature that is less than a threshold (0.5 using Taubin's curvature measure). This modification increases the correct match percentage to 91.1% and decreases matching time to 257 ms/image.

The final modification is to compress the spherical spin-images using principal component analysis [7]. For the test and the results presented in this paper, the spin-images were compressed from an image size of 50 bins to a spin-tuple dimension of 10; spin-tuples were compared using the l_2 norm. Although the correct matching percentage decreased to 88.0%, the decrease in matching time to 61 ms/image makes the slight loss worthwhile.

Taken as a whole, this sequence of improvements has increase the matching percentage to close to 90% (a 20% improvement) while decreasing the matching time to 61 ms/ image, a 600% improvement.

Table 1 Spin-image modifications.

Modification	% correct	Match time (ms)
Mesh interpolation	76.3	356
Spherical parameterization	80.8	336
Curvature Masking	91.1	257
Compression	88.0	61

2.3 Landmark Saliency

Surface symmetry or repetitive surface shape can cause spin-tuples from different places on a surface to be similar. This can cause landmark matching errors because, a scene tuple may match a model tuple very well even though the object-centered position of the landmarks is different. If spin-tuples that are similar to other spin-tuples can be detected and removed from the matching process, then the likelihood of correct matching will increase.

Saliency is a measure of how distinct a sample point is from other samples in its set. In matching, salient points are the points least likely to be confused with other points in the set. Saliency has been applied to the problems of face recognition [13] and stereo matching [8]. In our application, we would like to select the most salient spin-tuples on the model surface and use only those for matching. To do this we first need a definition and a way to compute spin-tuple saliency.

Saliency is inversely proportional to the density of the spin-tuple distribution. Under the assumption that a mixture of gaussians can model the spin-tuple distribution, we use the kernel method [2] to estimate the density at any point $\boldsymbol{x}$ in the spin-tuple space. At each spin-tuple $\boldsymbol{i}_i$ from the model containing N spin-tuples we place a Gaussian distribution G, the sum of which defines the spin-tuple density $p(x)$ at $\boldsymbol{x}$

$$p(\boldsymbol{x}) = \frac{1}{N}\sum_1^N G(\boldsymbol{x} - \boldsymbol{i}_i; \Sigma_i)$$

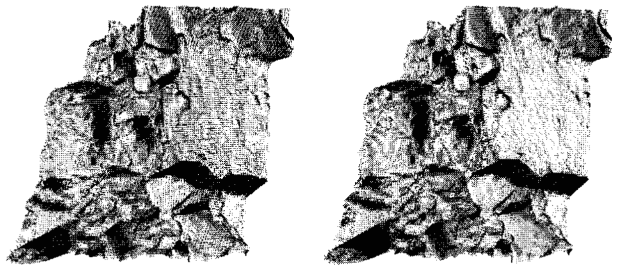
The covariance Σ_i is equal to the covariance of the entire distribution of spin-tuples sample up to a scale factor s. The covariance of the distribution is available from spin-image compression and the scale factor is set as suggested in [13] to

$$s = (\frac{4}{N}(t+2))^{1/(t+4)}$$

To compute the saliency of each spin-tuple we compute $p(\boldsymbol{i}_i)$. Spin-tuples with a density above a threshold are considered to have a saliency that is too low for accurate matching and are eliminated as landmarks.

Computing $p(\boldsymbol{i}_i)$ is an $\boldsymbol{O(n^2)}$ process, however it can be speed up considerably using the efficient closest point search structure described in the next section. Using this data structure all of the spin-tuples within a hypercube whose size is defined by the major axis of the spin-tuple covariance $\sum_i$ can be found efficiently. Then $p(\boldsymbol{i}_i)$ can be computed using just these spin-tuples and associated gaussians.

Figure 2 shows the landmarks on a surface before and after masking by saliency. Most of the landmarks in the flat part of the surface are eliminated because in flat areas, spin-tuples will be very similar. However, the spin-tuples were not eliminated in flat areas that are close to interesting surface features. This indicates that eliminating features based solely on flat curvature would not be appropriate because it would eliminate some landmarks that are distinct for matching and, being on flat parts of the surface, more likely to be accurate.



Original landmarks

Salient landmarks

Figure 2 Salient Landmarks.

3 Efficient Landmark Matching

Some knowledge of pose is usually available during landmark matching either from estimates derived from other sensors or problem context (e.g., a land vehicle does not drive upside down). By reducing the number of candidates during search for the best matching landmarks, this pose information can make landmark matching more efficient. It can also be used to reduce ambiguity between landmark matches; if multiple landmarks are similar then the closest one (in pose space) is chosen as the match. We use an efficient closest point search structure to store landmarks so that the incorporation of pose information into landmark matching is simple yet effective.

Before describing how pose constraints are used to limit search in landmark matching, we give an overview of our algorithm for landmark matching. In landmark matching for absolute position estimation, a small scene data set is usually searched for in a larger model data set. Model data is processed, before matching, to extract landmarks, which are then stored in an efficient closest point data structure as follows. First the model surface is interpolated and areas of high curvature are masked out. For the remaining mesh vertices, spherical spin-images are generated and these are then compressed into low dimensional spin-tuples using PCA. Surface landmarks (1) are then created by concatenating vertex position, surface normal and spin-tuples. These landmarks are then stored in the efficient closest landmark data structure. At runtime, a scene vertex is selected at random, and if it passes the curvature constraint, its spherical spin-image is generated using the model spin-image parameters. This spin-image is compressed using the model principal components and a scene landmark is generated from the scene vertex position, surface normal and spin-tuple. This landmark is then used to query the model closest landmark data structure to find the closest matching model landmark.

3.1 Closest Landmark Search

The closest landmark data structure used is a variant of the one first introduced in [9]. Their data structure enables efficient closest point search in high dimensional data using the simplification that if a closest point is not within a hypercube of size ε, it will not be found. The basic search principal of operation is as follows. First, model landmarks are inserted into the data structure, and sorted lists for each coordinate of the model landmarks are created. Arrays of forward and backward pointers keep track of the true and sorted location each of coordinate for each model landmark. At runtime, the model landmarks within a hypercube of size ε around the input scene landmark are found by first eliminating all model landmarks whose first coordinate is not within ε of the first coordinate of the scene landmark. The sorted lists of coordinates and the forward and backward pointers make this a logarithmic process for each coordinate. This process continues sequentially through the remaining coordinates; in the end, the model landmarks remaining are those that are within the hypercube. The closest model landmark is then selected as the remaining landmark that has the minimum l_2 distance to the scene landmark. The efficiency of search is controlled by ε; if ε is made small then more points are eliminated early in the search and the overall speed of search is increased.

In our approach we represent pose constraints as strict bounds placed on the translation and rotation angle between landmarks. A bounding box surrounding the scene landmark enforces the translational constraint; if a model landmark matches a scene landmark, its position is within the bounding box. We represent rotational constraints as a strict bound placed on the rotation angle between two landmarks. If two landmarks match then the rotation that aligns the surface normals of the landmarks is less than this angular bound. This angular bound can be transformed into a bounding box on surface normal coordinates.

With these insights, modification of the Nene and Nayar data structure to incorporate variable pose constraints into the landmark matching problem is straightforward. Instead of using a hypercube of scalar bound ε during closest point search, a hypercell with variable bounds

$$(2) \quad \varepsilon = (\varepsilon_x^p, \varepsilon_y^p, \varepsilon_z^p, \varepsilon_x^n, \varepsilon_y^n, \varepsilon_z^n, \varepsilon_i^i, \mathrm{K}, \varepsilon_t^i)$$

is used. Each of the pose dependent coordinates uses a different bound which, as explained below, depend on the constraints placed on the translation and rotation between landmarks. Landmarks are organized so that the pose dependent landmark coordinates (position and normal) are searched first during closest landmark matching. Consequently, landmarks that do not meet the pose constraints are eliminated in the closest point search structure before any invariant information is considered. After the pose dependent coordinates have been searched, the pose invariant coordinates are searched using a fixed bound. The remaining landmarks meet the pose constraints as well as the invariant constraint. Finally, the best matching model landmark is chosen from the remaining landmarks as the one that minimizes the l_2 distance between its invariant and the scene landmark invariant.

3.2 Setting Hypercell Bounds

To make the bounds explicit, suppose that the model landmark $\boldsymbol{l}^m$ matches a scene landmark $\boldsymbol{l}^s$

$$\boldsymbol{l}^s = [\boldsymbol{p}^s|\boldsymbol{n}^s|\boldsymbol{i}^s]^T = (p_x^s, p_y^s, p_z^s, n_x^s, n_y^s, n_z^s, i_1^s, \mathrm{K}, i_t^s)$$

$$\boldsymbol{l}^m = [\boldsymbol{p}^m|\boldsymbol{n}^m|\boldsymbol{i}^m]^T = (p_x^m, p_y^m, p_z^m, n_x^m, n_y^m, n_z^m, i_1^m, \mathrm{K}, i_t^m)$$

The strict hypercell bounds on translation between landmarks is established by the position bound ε^p

$$(3) \quad \left|p_i^m - p_i^s\right| \le \varepsilon^p \quad i = 1\mathrm{K}\,3$$

Because the bound on surface normal is expressed as a maximum angle α by which the surface normal can be rotated, defining the bounding box for surface normal is more involved. The basic idea is to construct the bounding box for surface normal as if it were the z-axis, given α, and then to rotate this bounding box to the landmark surface normal. A pictorial description of the process is given in Figure 1. The strict hypercell bounds on surface normal coordinates can be represented by the equation

$$(4) \quad \left|n_i^m - n_i^s\right| \le \varepsilon_i^n \quad i = 1\mathrm{K}\,3$$

We use a single bound on distance between invariants ε^i. We found that setting ε^i to the average distance between nearest neighbor model invariants produces good results. ε^i can be computed quickly by randomly querying the closest landmark data structure with a few existing model landmarks and taking the average distance between closest landmarks. The strict hypercell bounds on surface normal coordinates can be represented by the equation

$$(5) \quad \left|i_i^m - i_i^s\right| \le \varepsilon^i \quad i = 1\mathrm{K}\; t$$

It should be noted that the although the position and invariant bounds do not depend on the individual scene landmark position, the surface normal bounds do and must be computed anew for each scene landmark.

4 Robust Surface Matching

We match multiple scene landmarks to the model landmarks for the following reasons. First, a single landmark is not sufficient for computing a rigid transformation between model and scene; using position and normals, at least two landmarks must be matched. Second, if part of the model is occluded then the scene may contain landmarks that do not appear in the model. Finally, scene clutter and sensor noise can cause mismatches between model and scene landmarks. These mismatches need to be detected and removed so that they do not corrupt the final position estimate. Motivated by the Iterative Closest Point (ICP) algorithm [1][14] we have developed a robust iterative position estimation algorithm that handles multiple landmarks and eliminates landmark mismatches. An iteration has three stages: landmark matching, robust pose estimation and landmark transformation. We use an iterative algorithm because, it allows us to enforce geometric consistency between landmarks while computing the rigid transformation that aligns the scene to the model.

To match landmarks, first, multiple vertices are selected at random (on order 100) from the scene. If a selected vertex passes the curvature constraint, then its landmark is generated and the closest matching model landmark, which is within the pose constraints, is determined. This process is repeated for the remaining selected vertices, which generates multiple model landmark to scene landmark matches $[l_i^m, l_i^m]$.

Next, a robust Least Median Squares (LMedS) pose estimation algorithm is employed to remove mismatches and estimate pose. This algorithm investigates multiple triples of landmark matches to find a triple that is free of mismatches. The number of triples n is based on the expected percentage of mismatches o and the desired probability P of obtaining a sample without outliers.

$$n = \ln(1-P)/\ln(1-(1-o)^3)$$

For each triple, the best Least Squares (LSQ) rigid transformation *(R,t)* that aligns the scene landmarks to the matched model landmarks is computed [4]. Given this transformation the residual errors are computed for each landmark

$$r_i = \left\|\boldsymbol{R}\boldsymbol{p}_i^s + \boldsymbol{t} - \boldsymbol{p}_i^m\right\|^2 .$$

If the median residual error for these matches is less than the median residual error computed for all previous transformations r_{med}, the current transformation becomes the best encountered so far (R_{best}, t_{best}). The process is repeated for all of the n triples. Next the robust standard deviation

$$\sigma_r^2 = (1.4826(1+\frac{5}{n-3}))^2 r_{med}$$

is computed, and using σ_r, a landmark match is eliminated if

$$r_i = \left\| R_{best} \boldsymbol{p}_i^s + \boldsymbol{t}_{best} - \boldsymbol{p}_i^m \right\|^2 > \sigma_r$$

Using the remaining landmarks, a LSQ transformation is computed and it becomes the transformation for the current iteration.

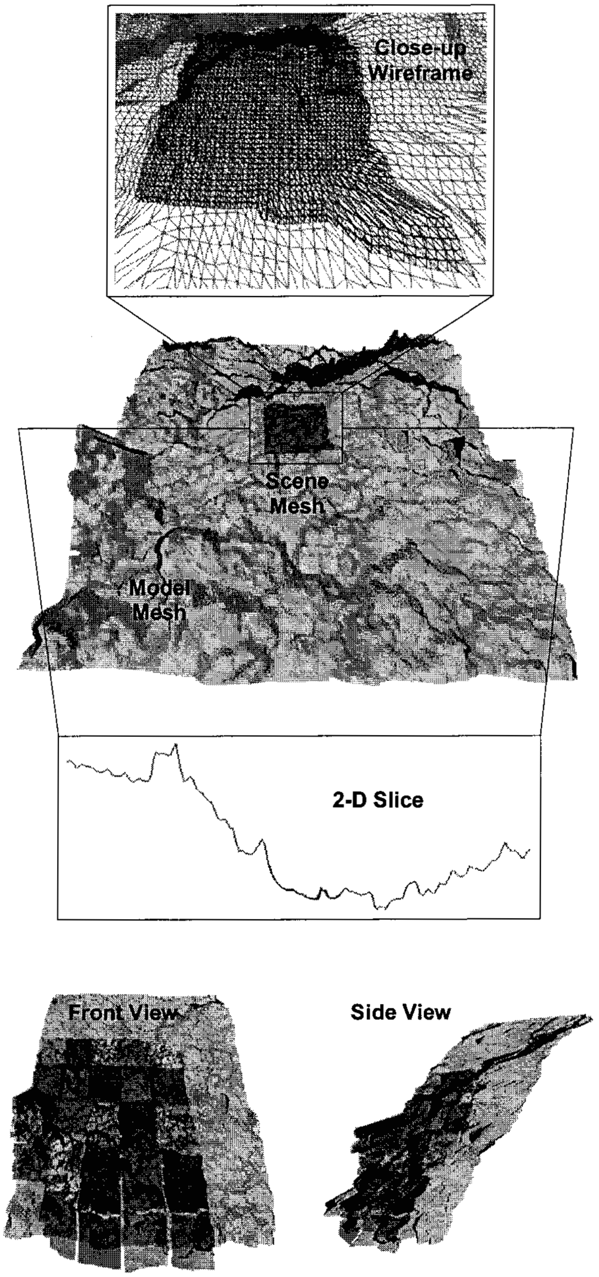
In the last step of each iteration the position and surface normal bounds are reduced. This is reasonable, because the scene surface is converging on the model and the distance between scene and model vertices as well as the angle between scene and model normals is decreasing. To set the new bounds on position and normal we use the algorithm presented by Zhang [14] for setting a threshold between points in his iterative closest point algorithm. This algorithm is based on constructing a histogram of the distances between matched points and from this histogram selecting a threshold that keeps point matches that are within the first mode of the histogram. This algorithm is applied directly to determine the new position bounds $(\varepsilon_x^p, \varepsilon_y^p, \varepsilon_z^p)$. This algorithm can also be applied to the matched surface normals if the angle between matched landmarks is used instead of the distance between vertices. The algorithm determines a new surface normal angle bound α which is then used to set the new bounds on surface normal $(\varepsilon_x^n, \varepsilon_y^n, \varepsilon_z^n)$.

As the iterations progress, the position and surface normal bounds decrease. This has the effect of enforcing the geometric consistency of matches. Since the distance between scene and model matches becomes smaller and smaller, the geometric configuration of model landmarks approaches the geometric configuration of the scene landmarks; this is the definition of geometric consistency.

5 Results

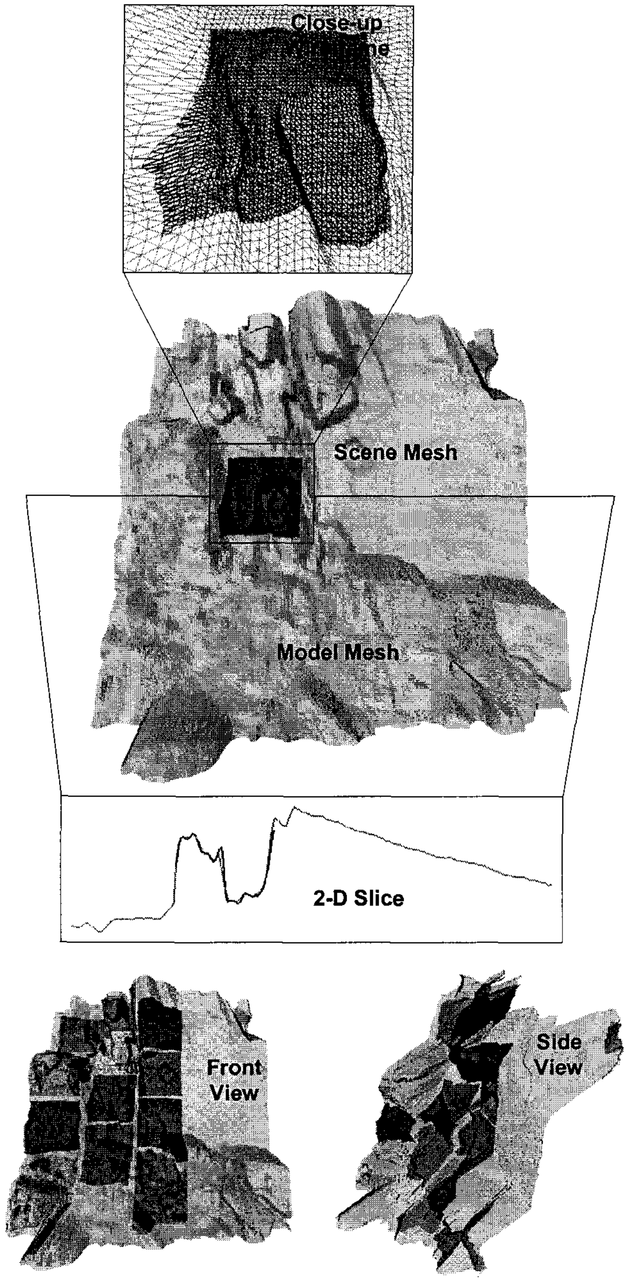
To test our algorithm, we collected various 3-D scans of natural terrain using a long range scanning laser rangefinder. The rangefinder has a maximum range of 800 m, a 10° field of view (FOV) with a programmable scan pattern that allows a maximum of 500x500 pixels to be collected.

The data sets for our first result were taken of a tree, rock and bush covered slope. Two scans were taken; one approximately 300 m from the slope and the other from a slightly different attitude and 250 m from the slope. Each scan had a 10° FOV and 500x500 samples. The ground truth change in position between the scans was determined through surveying of targets surrounding the sensor. To create the model mesh, the samples from the first scan were projected into a 10° FOV, 200x200 pixel range image; samples falling into the same pixel were averaged. From the range image a mesh was created. Similarly, to create each scene mesh, a 2° FOV, 50x50 pixel range image was created from a view slightly shifted from the sensor origin of the second scan. Using this procedure, each scene mesh covered approximately 4% of the model surface and the resolution of each scene mesh was 2x the resolution of the model.



Representative Scene and Model Matches

Figure 3 Surface matching result for a tree covered slope.



Representative Scene and Model Matches

Figure 4 Surface matching result for a rocky cliff..

We extracted 80 scenes from the second scan and successfully matched them to the model surface. Some representative scene meshes (color) are shown superimposed on the model mesh (gray) in Figure 3. As shown in the figure, scene meshes are only created in the region of the overlap between the scans. Figure 3 also shows a single scene mesh matched to a model surface, a close-up of the model and scene meshes showing the 2x difference in resolution and a 2-D slice cut horizontally though the two meshes showing the accurate alignment of the model and scene.

For these results we placed a strong bounds on rotation between landmarks (α=10°) and no constraint on the position between the landmarks. These constraints would be typical of a sensor platform equipped with gyros.

Typical registration and timing result are given in Table 2. The table shows that the registration errors are within the resolution of the meshes being registered and that the entire matching process including landmark generation, matching and alignment takes less than 20 seconds on a 174 MHz R10000 SGI O2 workstation. This result shows that we can rapidly match small scenes to a large model with a 2x difference in resolution.

Figure 4 shows a result for two scans taken of a rocky cliff. In this result, the two scans are taken with a lateral shift of 0.6 m between scans and the cliff face is approximately 16 m away. 49 scenes were generated from the second scan in exactly the same way as explained above. All but three of the scenes were correctly matched to the model. The scenes that were not matched correctly corresponded to the flat region on the right side of the model. This region (as shown in Figure 2) lacked adequate salient landmarks, so matching enough landmarks in this region was not possible. As in the above result, no bound was placed on translation and a small (10°) bound was placed on the maximum rotation between landmarks. As shown in Figure 4, the alignment of the surfaces is quite accurate. However, Table 2 shows that the error in the absolute translation and rotation are larger than expected. The cause of this is inadequate survey data to accurately estimate ground truth. The final result shows the stitching together of 13 scans of a rocky slope taken as the sensor was mechanically panned. The scans were taken 5° apart and each had a 10° FOV and 500x500 samples. Each scan was projected into a 200x200 range image with a 10° FOV and the landmarks from adjacent scans were matched. During matching, a 35 m position bound and a 30° surface normal bound was placed on the landmarks. Knowledge of pan angle was not used to initialize the rotation between views. Figure 5 and Table 2 shows that this type of panoramic data can be accurately and rapidly matched. This result also demonstrates the accurate matching of two surfaces that have only 50% overlap.

6 Conclusion

We have presented and algorithm for robust and efficient surface landmark matching. Our algorithm is based on intelligent landmark selection and the incorporation of pose information into the landmark matching process. We have shown results where a small scene patch is matched to a coarse model. This scenario is particularly difficult because of the difference in resolution between the model and scene and the relatively small size of the scene relative to the model. In the future we plan to apply this work to self-localization of autonomous aerial vehicles and precision landing on comets and asteroids.

References

[1] P. Besl and M. McKay. "A Method for Registration of 3-D Shapes." *PAMI*, 14 (2), pp. 239-256, 1992.

[2] C. Bishop, Neural Networks for Pattern Recognition, New York: Oxford University Press, 1995.

[3] O. Carmichael and M. Hebert, "Unconstrained Registration of Large 3D Point Sets for Complex Model Building," Proc. IEEE/RSJ Int'l Conf. On Intelligent Robotics and Systems (IROS98), pp. 360-366, 1998.

[4] O. Faugeras and M. Hebert. "The Representation, recognition and Locating of 3D Shape from Range Data." *IJRR*, 5 (3), pp. 27-52, 1986.

[5] D. Huber and M. Hebert, "A New Approach to 3-D Terrain Mapping," Proc. IEEE/RSJ Int'l Conf. On Intelligent Robotics and Systems (IROS99), pp. 1121-1127, 1999.

[6] A. Johnson and M. Hebert. "Surface Matching for Object Recognition in Complex Three-Dimensional Scenes," Image and Vision Computing, 16, pp. 635-651, 1998.

[7] A. Johnson and M. Hebert, Using Spin-Images for Efficient Object Recognition in Cluttered 3-D Scenes," IEEE Trans. Pattern Analysis and Machine Intelligence, 21(5), pp. 433-449, 1999.

[8] R. Manduchi and C. Tomasi, "Distinctiveness Maps for Image Matching," 10th ICIAP, September 1999.

[9] S. Nene and S. Nayar, "A Simple Algorithm for Nearest Neighbor Search in High Dimensions," IEEE Trans. Pattern Analysis and Machine Intelligence, 17(9), pp. 989-1003, 1999.

[10] G. Sharp, S. Lee and D. Wehe, "Invariant Features and the Registration of Rigid Bodies," IEEE Int'l Conf. Robotics and Automation (ICRA'99), pp. 932-937, 1999.

[11] G. Taubin, "Curve and Surface Smoothing Without Shrinking," Proc. IEEE Int'l Conf. Computer Vision (ICCV95), pp. 852-857, 1995.

[12] G. Taubin, "Estimating the Tensor of Curvature of a Surface from a Polyhedral Approximation," Proc. IEEE Int'l Conf. Computer Vision (ICCV95), pp. 902-907, 1995.

[13] K. Walker, T. Cootes and C Taylor, "Locating Salient Features Using Image Invariants," Proc. Int'l Conf on Automatic Face and Gesture Recognition, pp. 242-247, 1998.

[14] Z. Zhang. "Iterative Point Matching for Registration of Free-Form Curves and Surfaces." *IJCV*, 13 (2), pp. 116-152, 1994

Acknowledgements

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. I would like to thank Miguel SanMartin, Fred Hadaegh, Aron Wolf for supporting this work and Curtis Padget and Dan Clouse for technical support during data collection.

Table 2 Absolute position estimate results and parameters.

Result	True Translation	True Rotation Angle	Translation Error	Rotation Angle Error	Model Resolution	Scene Resolution	Landmark Generation Time	Match Time	Align Time
Slope	50.0 m	2.2°	0.59 m	0.19°	0.45 m	0.19 m	11.4 s	4.2 s	4.4 s
Cliff	0.6 m	0.3°	0.16 m	0.61°	0.32 m	0.17 m	10.9 s	2.7 s	4.2 s
Panorama	0.0 m	5.0°	0.12 m	0.10°	0.25 m	0.35 m	23.0s	2.4 s	8.2 s

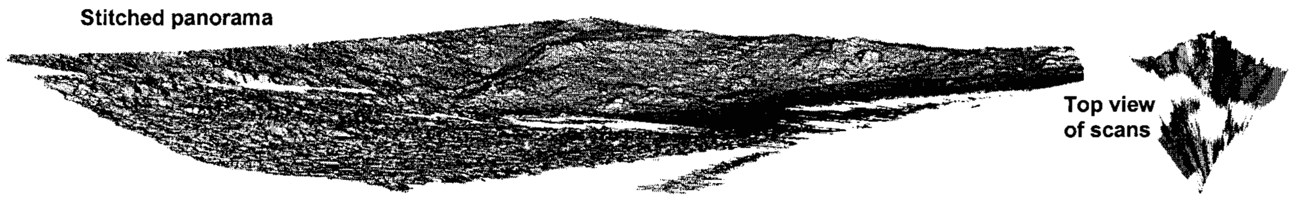


Figure 5 Surface matching result for a panorama of images. 13 scans were taken with 5°pan between scans.